



# Java Graphics on Mac OS X

**Session 401**





# Java Graphics on Mac OS X

**Gerard Ziemski**  
**Java Classes Engineer**

# Session Overview

- Java Graphics on Mac OS X
- Advanced Topics
- Future





# Java Graphics on Mac OS X

# Quartz-based Graphics

- Standard `java.awt.Graphics` features
- Advanced `java.awt.Graphics2D` features
- Beyond Java2D



# Graphics

- Basic Primitives: lines, rectangles, ovals, polygons
- Color
- Basic Images
- Text



# Graphics2D

- Highly extensible
- Low level
- Fully featured



# Graphics2D

- Understanding Java2D rendering model:
  - Graphics Contexts
  - Graphics Contexts Attributes
  - Graphics Objects
- Concepts required for making the most out of Graphics2D
  - BufferedImages
  - Transformations





# Graphics2D

## Java

## Quartz

---

Shape OBJECTS

CGContextMoveToPoint,  
CGContextAddQuadCurveToPoint, . . .

---

Compositing

kCGSCompositeSover,  
kCGSCompositeSin, . . .

---

Strokes

CGContextStrokePath,  
CGContextSetStrokePattern, . . .

---



# Beyond Java2D

- Double Buffered Contexts
- PDF underpinnings
- Font support





# Advanced Topics

# Advanced Topics

- Rendering hints
- Graphical Contexts
- Graphics hardware acceleration
- Hints and tips



# Rendering Hints

- Optional
- Default settings different than on other platforms
- Setting hints using options (affects Aqua L&F)
  - Text rendering hints
    - `com.apple.macosx.AntiAliasedTextOn`
  - Graphics rendering hints
    - `com.apple.macosx.AntiAliasedGraphicsOn`
- Setting hints using graphics object
  - `g2.setRenderingHint`
- Other rendering hints



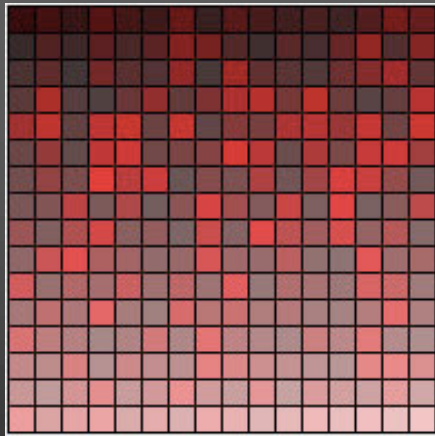
# Graphical Contexts

- Two types of contexts, based on their function
  - Source
  - Destination
- Several types of context, based on their pixel representation
  - RGB
  - ARGB
  - BGR
  - BGRA



# Graphical Contexts

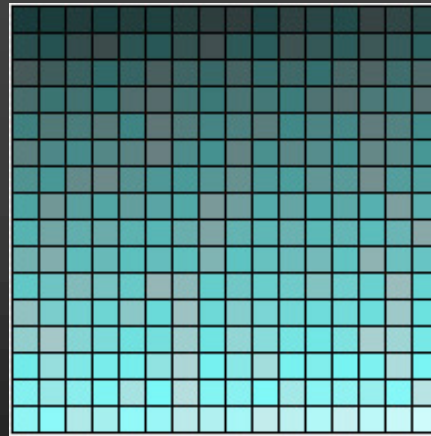
**Quartz**



Window  
(screen)

native Quartz surface

**Java**



Graphics  
Context (dst)

Java surface

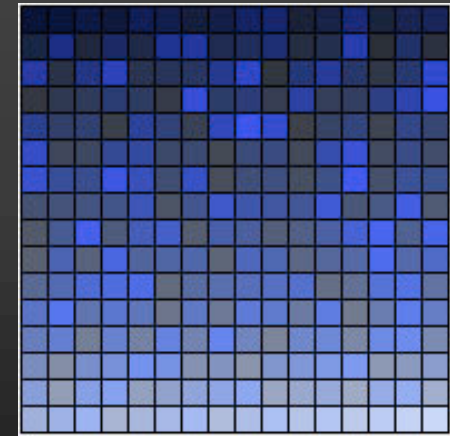


Image (src),  
Graphics primitives



**Flow of pixels**



# Graphical Contexts

- Slow animation:

```
pixels = new byte[w*h];
...
source = new MemoryImageSource(w,h,icm,pixels,0,w);
source.setAnimated(true);
source.setFullBufferUpdates(true);
img = createImage(source);
...
icm = new IndexColorModel(8, 256, paletteTable[0], paletteTable[1],
paletteTable[2]);
...
pixels[index++]= (byte)result;
```





# Graphical Contexts

- Fast animation:

```
GraphicsEnvironment ge = GraphicsEnvironment.getLocalGraphicsEnvironment();
GraphicsDevice gd = ge.getDefaultScreenDevice();
GraphicsConfiguration gc = gd.getDefaultConfiguration();
ColorModel cm = gc.getColorModel();
WritableRaster raster = cm.createCompatibleWritableRaster(w, h);
db = raster.getDataBuffer();
bim = new BufferedImage(cm, raster, cm.isAlphaPremultiplied(), null);
...
pixels = (((DataBufferUShort)db).getData());
...
if (db.getDataType() == DataBuffer.TYPE_INT)
    pixels[index++] = 0xff000000 | (red<<16) | (green<<8) | blue;
```



# Hardware Acceleration

- Introduction
- Quality vs. speed
- Activation
- Heuristic and fallbacks
- UI Framework implications





# Demo

**Hardware Acceleration**

# Hints and Tips

- Use hwaccel if possible
- Use appropriate image types
- Use Clipping
- Cache fonts objects
- Do not double buffer
- Do not mix AWT and Swing





The Future

# JDK 1.4

- Pluggable Image I/O Framework
- Better fonts/text support
  - TrueType hinted fonts
  - Unicode 3.0 Bidi algorithm
- Full Porter-Duff Compositing Rules Support



# JDK 1.4

- New pipeline architecture
- VolatileImages
  - Hardware accelerated offscreen contexts
  - Offer fast path for images (preferably static)
  - Support basic 2D operations
  - Cache glyphs for high text performance



# 3D Graphics

- Java3D
- gl4java







# Demo

**1.4 Native IO/gl4java**

**Kenneth B. Russell**  
**JavaVM Engineer at Sun Microsystems**

# Where to Go From Here

---

**402 J2SE Mac OS X  
Platform Overview**

Hall 2  
**Tues., 2:00pm**

---

**404 Tailoring Java Applications  
for Mac OS X**

Room J  
**Tues., 5:00pm**

---

**505 OpenGL: Integrated Graphics I**

Room J  
**Wed., 9:00am**

---

**506 OpenGL: Integrated Graphics II**

Room J  
**Wed., 10:30am**

---



# Where to Go From Here

---

**514 OpenGL: Performance  
and Optimization**

Room J  
**Thurs., 5:00pm**

---

**407 Java Performance**

Room C  
**Fri., 9:00am**

---

**516 Graphics and Imaging  
Performance Tuning**

Hall 2  
**Fri., 3:30pm**



# Who to Contact

---

**Alan Samuel**

Java Technologies Evangelist

**[bluker1@apple.com](mailto:bluker1@apple.com)**

---

<http://developer.apple.com/wwdc2002/urls.html>



# Links and Resources

- Apple Java Developer Resources
  - <http://developer.apple.com/java>
    - New Documentation!
- Java-dev Mailing List
  - [java-dev@lists.apple.com](mailto:java-dev@lists.apple.com)
- Developer Technical Support
  - [dts@apple.com](mailto:dts@apple.com)



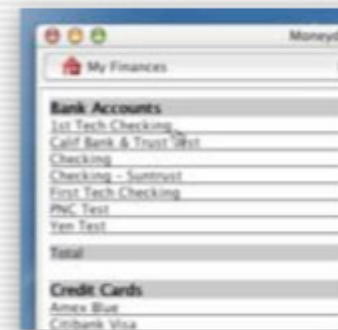
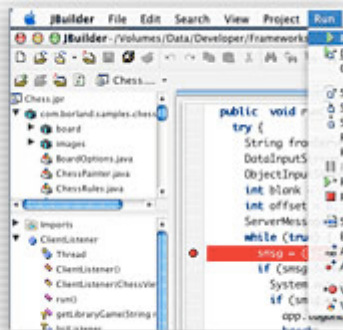
# How to Access Documentation

- Most up-to-date: PDF and HTML  
<http://developer.apple.com/techpubs/java>
- Hardcopy print-on-demand  
[Vervante.com](http://Vervante.com) under Related Resources
- Product CD  
Documents folder and installed in  
[/Developer/Documentation/Java](#)
- Check ADC News for latest updates  
<http://developer.apple.com/devnews>





# Q&A



**Alan Samuel**  
**Java Technologies Evangelist**  
**bluker1@apple.com**

<http://developer.apple.com/wwdc2002/urls.html>

 **WWDC2002**



 **WWDC2002**

 **WWDC2002**